End-User Development Success Factors and their Application to Composite Web Development Environments

David Lizcano, Fernando Alonso, Javier Soriano and Genoveva López
DLSIIS, Dept. of Computer Science
Universidad Politécnica de Madrid
Madrid, Spain
Email: dlizcano@fi.upm.es

***Abstract*—The Future Internet is expected to be composed of a mesh of interoperable Web services accessed from all over the Web. This approach has not yet caught on since global user-service interaction is still an open issue. Successful composite applications rely on heavyweight service orchestration technologies that raise the bar far above end-user skills. The weakness lies in the abstraction of the underlying service front-end architecture rather than the infrastructure technologies themselves. In our opinion, the best approach is to offer end-to-end composition from user interface to service invocation, as well as an understandable abstraction of both building blocks and a visual composition technique. In this paper we formalize our vision with regard to the next-generation front-end Web technology that will enable integrated access to services, contents and things in the Future Internet. We present a novel reference architecture designed to empower non-technical end users to create and share their own self-service composite applications. A tool implementing this architecture has been developed as part of the European FP7 FAST Project and EzWeb Project, allowing us to validate the rationale behind our approach.**

***Keywords*-End-User Development, User-Centred Service-Oriented Architectures, Service Front-Ends, Composite Applications, Future Internet, Internet of Services**

I. Introduction

Service-Oriented Architectures (SOA) have attracted a great deal of interest over the last few years. In fact, SOAs increase asset reuse, reduce integration expenses and improve business agility in responding to new demands [1].

Nonetheless, mainstream development and research into SOAs have until now focused mainly on middleware and scalability, service engineering and automating service composition using business modelling process (BPM) technologies. Little or no attention has been paid to service front-ends, which we view as a fundamental part of SOAs [2]. As a result, SOAs remain on a technical layer hidden away from the end user.

The evolution of Web-based interfaces bears testimony to the progress made towards improving service usability. However, existing, web-based service front-ends do not come at all close to meeting end-user expectations [3]. Applications and information portals are still based on monolithic, inflexible, non-context-aware, non-customizable and unfriendly user interfaces (UIs). Consequently end users do not really benefit from the advantages promoted by service orientation in terms of modularity, flexibility and composition [4]. In addition service front-ends are constructed ad hoc without formal engineering methods and tools that could accelerate the time to market.

The vision presented here is an early result of the Service Front End (SFE) Open Alliance initiative[1]. This initiative aims to integrate results from several relevant R&D projects in the field to produce open specifications and an open source reference implementation of components of an envisioned Web platform to access services, contents and things in the Future Internet. This would enable end-user development (EUD) of software solutions based on user-centred services. The SFE Open Alliance initiative was setup under the umbrella of activities within the Service Front Ends Collaboration Working Group created in the FP7 call and currently involves projects such as FAST or EzWeb.

Section II of the paper states the service front-end problem. Section III presents a framework for studying EUD success in current solutions in order to elicit vital guiding principles to drive our search for the shortcomings of existing service front-end technology. Then Section IV proposes a novel reference model and architecture that empowers end users and supports this vision. This architecture will enable the creation of new ecosystems, where all stakeholders will be able to collaboratively develop capabilities and innovate new operating procedures by mixing and integrating already available services. Then, the above ideas are briefly validated in Section V. Finally, Section VI discusses the main conclusions of this research.

II. Shortcomings on the Road Towards an Internet of Services enabling EUD

The provision and consumption of information-intensive electronic services across corporate boundaries has attracted considerable interest over recent years. Particularly the Web services technology stack [1] was expected to act as efficient and agile "plumbing [...] for information systems to interact

[1]Open Alliance for Service Front Ends, http://sfe.morfeo-project.org

without human involvement" [5]. Following the design principles suggested by SOAs, Web services provide a uniform, system-independent way for interlinking dispersed electronic services. While technology and standards are important for achieving the vision of a globally networked, agile service economy, it has been widely recognized today that they are not sufficient on their own [3]. Analyses of today's cross-organizational service interconnections following the SOA paradigm have resulted in the identification of the following major weaknesses:

- ***Rigid and process-oriented composition***. Not all the potential of SOAs has been unleashed yet. Adherence to merely process-oriented design principles leads to rigid applications that cause huge reprogramming efforts in the event of changes. As in the 1970s, where the prevalence of Spaghetti-code-like software programming led to unmanageability and unchangeability of applications (the software crisis [6], [7]), the application of inflexible service orchestration techniques (e.g., based on BPEL (Business Process Execution Language)) prevents SOAs today from being truly agile.
- ***Deficient interoperability***. A second major issue of today's SOAs concerns service interoperability. Sometimes referred to as the "corporate household problem", information objects defined as input or output messages of services are based on highly proprietary specifications. The resulting semantic and syntactical heterogeneity causes significant mapping efforts when different services are to be interconnected and often leads to errors and increased costs.
- ***Limited retrievability***. In today's Internet, a lack of comprehensive, trustworthy and widely accepted service registries is another roadblock on the way to a networked service economy. In fact, a number of intermediaries are required to provide rich navigation to users, as well as to improve transparency and thus fulfil institutional functionality.
- ***Mute and autistic service interfaces***. Technologies such as the above Web services stack aim at supporting the setup of loosely coupled application interconnections especially in a professional context and assume users to be technically qualified. WSDL-based interfaces, for example, do not allow for rich interaction between machines and human users, but rather focus on automated machine-to-machine interoperation.

All these weaknesses prevent current services from being really useful for non-tech end users, who are unable to easily exploit them to develop and compose their own solutions. Therefore, Section III will review and analyse major current EUD solutions with the aim of exploring the design principles, factors and issues to be dealt with in order to achieve EUD success.

III. Existing successful EUD solutions

Nowadays there are several applications empowering end users without programming skills to develop their own software solutions, fitted to their unique and instant requirements. These applications, like spreadsheets, e-mail filter creators or mashup Web platforms, focus on outputting different software solutions, each oriented to a specific problem domain, such as calculation requirements, spam filtering or visual Web widget composition.

Of all these approaches, several studies state that spreadsheets are the most relevant and successful EUD solution existing at present [8]. In an empirical study carried out by Wu et al. (2007), 100% of a huge sample of end users had at some time used a spreadsheet program in their daily work to solve some problem or other. Other publications, like Boehm et al. (2005), establish that more than 55 million people in the United States do this kind of EUD programming, whereas professional programmers account for about 2.75 million of the country's population. This gap is actually widening, as Scaffidi et al. (2005) predict that the EUD population (users of spreadsheets and other EUD approaches) at workplaces in the United States will be 90 million by 2012. These studies and publications indicate that EUD is about to take control not only of personalizing computer applications and writing programs but of designing new computer-based applications without ever seeing the underlying program code.

For this reason many researchers around the globe have begun to study EUD success factors, focusing above all on the most relevant EUD solution, spreadsheets, to understand which of their principles and factors are used and accepted by end users. Studies like [8] focus on successful human EUD factors, since other studies like [9] focus on HCI (human-computer interaction) factors or on aspects of specialization and functionality [10].

The feedback that we get after reviewing all referenced studies is that EUD success is related to *human factors*, *HCI factors* and the *specialization-functionality relationship*. However, there are no publications that put all these ideas together to offer a general conception of EUD success factors. In this paper we will review the most relevant publications and scientific results in the EUD domain and then take a step further by combining them in an innovative EUD success framework. This way, we will be able to study each EUD solution and form a general idea of how successful it is likely to be among end users based on the studied factors. Additionally this framework will be useful for obtaining which design and architectural decisions are relevant for achieving end-user satisfaction, something that is vital for improving future EUD solutions such as the one proposed in this paper and fostering EUD success.

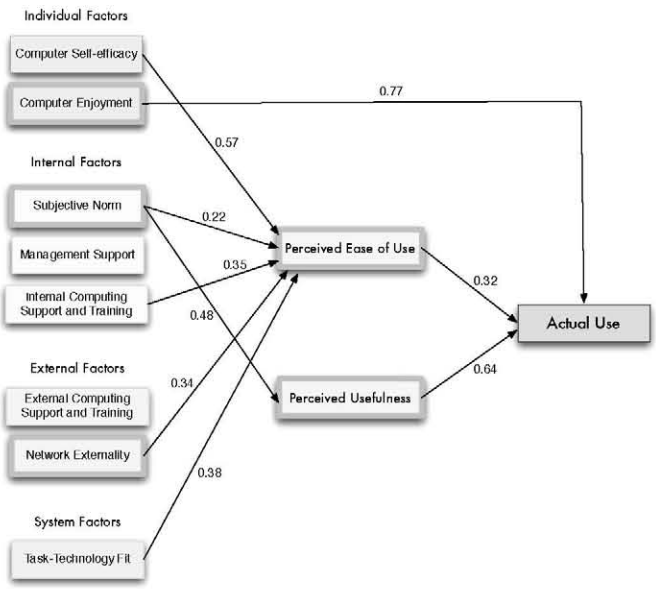

Figure 1. Empirical results of study about human factors related to EUD success

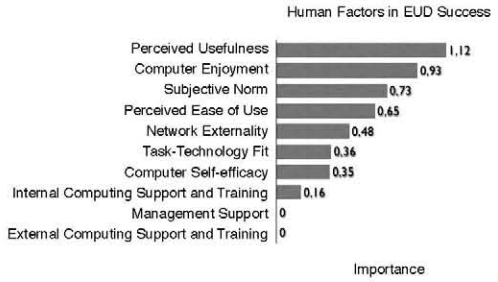

Figure 2. Importance of each human factor related to EUD success

A. *Successful Human Factors*

All software development tools should be well accepted by their target users if they are to be considered a successful solution. However, this acceptance, as Wu et al. (2007) show in their empirical survey, is not down to the choice of particular technologies or architectural decisions, but because they preserve and take care of a number of human factors.

End-user computer acceptance (EUC) has been established by Wu et al. as one of the critical success factors in achieving business success, and is defined as the adoption and use of information technology by personnel outside the IT domain to develop software applications in support of organizational tasks. Davis [11] proposed the technology acceptance model derived from the reasoned action theory that has been tested and extended by numerous empirical researchers. In these studies the actual use of any application is derived from several human factors as perceived ease of use, perceived usefulness, and so on, and these ideas were the basis for Wu et al.'s research.

The empirical study carried out by Wu et al. relied on 800 people testing programs and evaluating software solutions. The evaluation showed that actual software use follows the causal relationships illustrated in Figure 1. This diagram establishes what factors are related to the actual end use, and what is the weight or strength of this causal relation, expressed by a correlation coefficient. The most relevant factors are explained in detail below.

- Perceived Ease of Use: The degree to which a person believes that using specific software would be effortless.
- Perceived Usefulness: The degree to which a person believes that using specific software will increase his or her job performance.
- Computer Self-Efficacy: A person's perception of their ability to use computers to complete a task.
- Computer Enjoyment: Individuals that experience immediate pleasure and joy from using software.
- Subjective Norm: The degree to which a person believes that people that are important to him or her think that he or she should do the thing in question.
- Internal Computing Support and Training: Technical support and the amount of training provided inside the company.
- Network Externality: The utility of software use increases if the number of users increases.
- Task-Technology Fit: The degree to which an organization's application meets the information needs of the task.
- Management support or external training provided from outside the company is not related to end use according to this study.

In Figure 1 there are factors with multiple weighted paths to the final concept "actual use", and therefore the correlation coefficient between each factor and the use of a program is not clear. In Figure 2 all correlation coefficients have been recalculated in order to show the final impact of each factor on the actual use of the software. This way, it is possible to scale every factor and get an idea of its relevance.

This study suggests that an end user will use a program if he or she perceives it to be useful and enjoys the experience of using it. If an application is used in the end user's environment, it is more natural for him or her to accept and use this software too. Finally, ease of use and the fact that application usefulness would increase when it is used by more and more users will cause more actual use of a software tool.

B. *Successful HCI Factors*

Other studies like Jones et al. (2003) claim that spreadsheets (and other similar EUD solutions) are the programming language of choice for many people because

of their human-computer interaction facilities. Spreadsheets are a user-centred approach to language design, focusing on fostering usability through effective human-computer interaction. Specialized research into the psychology of programming and empirical studies of programmers [9] offer a groundwork for human issues in programming, structured as cognitive dimensions, that EUD solutions should consider and optimize in order to be successful among end users. These cognitive dimensions prove to be HCI factors that, if properly taken care of, result in high end-user acceptance on a par with spreadsheets. Green and Petre (1996) defined 13 cognitive dimensions (all of which were of equal importance) that, if well looked after, improve HCI and simplify EUD [12]. The most quoted of these factors are listed below:

- Abstraction gradient: What are the minimum and maximum levels of abstraction? Can fragments be encapsulated?
- Consistency: When some of the language has been learnt, how much of the rest can be inferred?
- Error-proneness: Does the design of the notation induce "care-less mistakes"?
- Hidden dependencies: Is every dependency overtly indicated in both directions? Is the indication perceptual or only symbolic?
- Premature commitment: Do programmers have to make decisions before they have the information they need?
- Progressive evaluation: Can a partially complete program be executed to gather feedback on "How am I doing"?
- Role expressiveness: Can the reader see how each component of a program relates to the whole?
- Viscosity: How much effort is required to make a single change?
- Visibility and juxtaposability: Is every part of the code simultaneously visible, or is it at least possible to compare any two parts side-by-side at will? If the code is dispersed, is it at least possible to know in what order to read it?

According to Jones et al.'s study, software that looks after these factors, like Microsoft Excel or other spreadsheet solutions, enable users without programming skills to implement software in a simple and flexible manner. Therefore, these dimensions must be kept in mind when new EUD approaches are set out.

C. Successful Specialization/Functionality Trade-off

For many researchers in the EUD and composite applications domain, the most relevant factor for success among end users is that EUD software accomplishes a good trade-off between the specialization and the functionality of the created solutions [10]. This relationship gives an idea about whether end users could create their own solutions to satisfy their needs. How well suited a developed solution is for a task or real problem could be quantified by two factors:

- Specialization: the degree to which an application exactly matches real requirements, features and details of a real problem.
- Functionality: the sum or any aspect of what a product, such as a software application or computing device, can do for a user. The overall functionality decreases when the solution is overly specialized for a specific problem.

However, these factors are opposite. It is impossible to increase one factor without decreasing the other. For this reason, EUD solutions should adopt a trade-off where both factors are at equilibrium. This will lead to solutions that are very specialized for a problem but could be easily exported and used in other problem domains. This balance is frequently measured on a four-point Likert scale (poor, average, good or optimal specialization/functionality relationship) [10].

D. Framework for Studying and Eliciting key EUD Success Factors to Improve EUD Solutions

All the factors explained above are frequently referenced and used in EUD research, but they are always applied individually. In this paper we propose the creation of a complex framework to study key EUD success factors globally by combining all the studied factors.

First of all, they all have to be compared to study interrelations. Because each factor type focuses on one vertex of the HCI domain (human, human and computer interaction and software), we must conclude that human factors, HCI factors and the specialization/functionality factor are orthogonal to each other. Bearing this premise in mind, it is possible to join each family of factors as an independent axis in a 3D plot that represents each independent family of factors in a visual manner (Figure 3). Each axis must be managed as follows:

- X-axis = human factors. In the last section we described eight factors that should be considered to achieve EUD success. These factors had correlation coefficients to indicate their relevance. In the proposed framework, the study of an EUD solution will include an evaluation of every factor according to a three-point Likert scale (0 for low factor rating, 1 for an average rating and 2 for a high rating). This rating will be multiplied by the correlation coefficient (shown in Figure 2) to output a final rating for this factor. Every factor must be evaluated and added together for each EUD solution studied. This will add up to a final rating of from 0 to 9.56 (due to correlation coefficients). Finally, this rating has to be normalized to a standard scale ranging from 0 to 10 (by multiplying by 10/9.56). This final value will be represented on the X-axis and give visual information about how successful the studied solution would be in terms of EUD based on human factors.
- Y-axis = HCI factors. In the previous section we studied thirteen HCI factors that should be improved to achieve

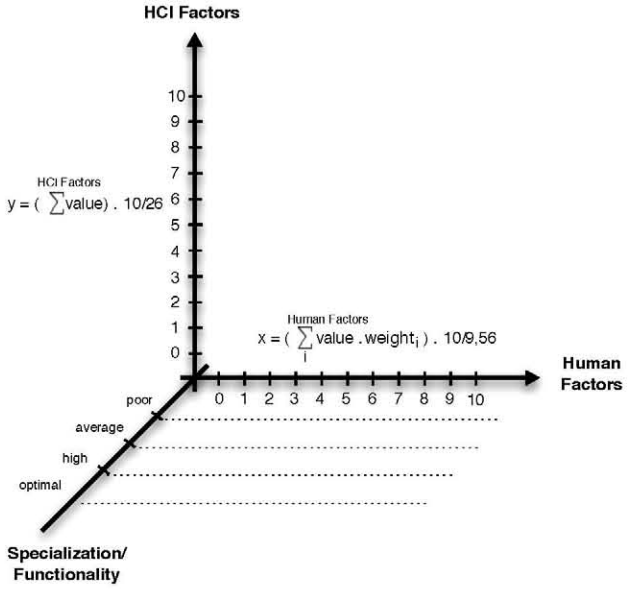

Figure 3. Framework of key EUD success factors

EUD success, all of which were equally important. Therefore, an analysis of these factors for an EUD solution will involve evaluating each factor on a three-point Likert scale (0 for a low rating, 1 for an average rating and 2 for a high rating of the factor) and add up this value for each factor. This process will output an overall rating ranging from 0 to 26 points. Finally, this value has to be normalized to an understandable scale (0-10) by multiplying by 10/26. This final value will be represented on the Y-axis giving a visual idea about how successful the studied solution would be in terms of EUD based on its cognitive dimensions.

- Z-axis = Specialization/functionality trade-off: the four-point Likert score studied in Section III-C could be represented directly on the z-axis, adding a third dimension to the solution's expected EUD success.

This framework is very useful in two ways:

- It is a powerful tool for studying any EUD solution and forming an idea of expected EUD success based on extended and proven principles, founded on factors included in several research papers.
- This framework summarizes all proven factors that are related to actual use and user acceptance, so it is a good starting point for creating new EUD environments or approaches.

Starting from referenced studies about spreadsheets and specific solutions like Excel, we can exploit user impressions and evaluations [8], [9], [10] to plot the expected success of Excel using the presented framework. This way, we could observe the performance of a successful EUD solution in the framework, giving an idea of what is the ultimate goal when new EUD solutions are to be developed. Table 4 presents a

Human Factors	Value	Adjusted Value
Perceived Usefulness	2	2,24
Computer Enjoyment	1	0,93
Subjective Norm	2	1,46
Perceived Ease of Use	2	1,3
Network Externality	2	0,96
Task-Technology Fit	1	0,36
Computer Self-efficacy	2	0,7
Internal Computing Support and Training	2	0,32
Total		***8,65***
HCI Factors	**Value**	
Abstraction Gradient	1	
Closeness of Mapping	1	
Consistency	2	
Diffuseness	2	
Error-proneness	2	
Hard Mental Operations	1	
Hidden Dependencies	2	
Premature Commitment	2	
Progressive Evaluation	2	
Role-Expressiveness	2	
Secondary Notation	2	
Viscosity	1	
Visibility	2	
Total	***8,46***	
Specialization/Functionality Factor	**Value**	
Total	***Average***	

Figure 4. Excel evaluation from studies [8] and [9]

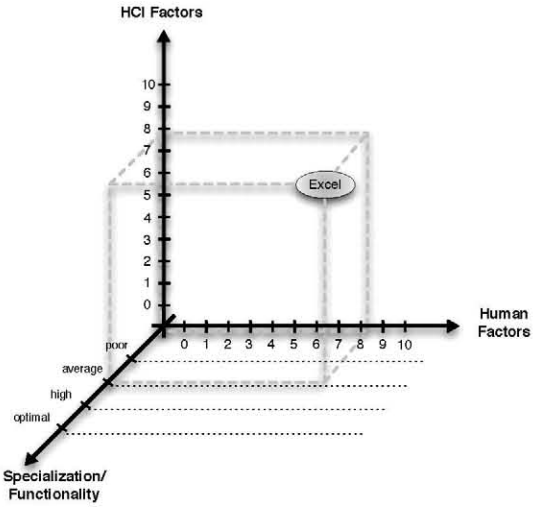

Figure 5. Excel EUD success represented in the proposed framework

user evaluation of Excel, whereas the final rating of Excel is shown in Figure 5.

E. Guiding EUD Principles Enabling the Internet of Services

From our point of view, the factors and principles explained previously state the need for user-centric SOAs based on a new generation of service front-end technologies in order to achieve EUD success through the Internet of Services. Such technologies will enable the massive deployment

of services on the Internet, driven by the following guiding principles:

- ***End-User Empowerment***, enhancing traditional user-service interaction by facilitating the selection, creation, composition, customization, reuse and sharing of applications in a personalized operating environment [13].
- ***Seamless Context-Aware User-Service Interaction***. New-generation service front-ends should have the capability to detect, represent, manipulate, and use contextual information to adapt seamlessly to each situation, supporting human users in a more effective, personalized and consistent way [3]. Novel engineering tools and methods should be devised in order to support context-aware service front-ends.
- ***End-User Knowledge Exploitation***. This principle aims to exploit users' domain knowledge and collective intelligence to improve service front-ends. End users' knowledge can be used to tag resources using light semantics, assist while interacting with services, enrich contextual information (e.g. by means of automatic user profiling) and infer new candidate processes to be later automated (on the back-end) [5].
- ***Universal Collaborative Business Ecosystems***. Enterprise systems should incorporate advanced user-centric, context-aware front-ends to enable their employees and other stakeholders to exploit and share their extensive domain expertise, and their thorough business knowledge [4]. Employees, customers, developers and providers will collaborate to create and improve enterprise applications, sharing, reusing, changing and combining existing context-aware components (services, contents, things...)[14].

The EUD solution presented in this paper has been conceived following the principles and ideas presented in the EUD success framework in order to procure as much end-user acceptance as possible. Finally, our approach is evaluated in the validation section to study its success and compare it with spreadsheet solutions.

IV. Materializing EUD Principles and Success Factors to Reach a User-Centred Internet of Services

In this section we propose a novel architecture for next-generation service front-ends. This architecture was devised in accordance with the presented guiding principles, and applies human, HCI and functionality/specialization factors studied in Section III. This whole complex architecture is being researched and developed as part of several major R&D&i projects like NEXOF-RA, EzWeb, MyMobileWeb and FAST, and its objective is that end-users with no programming skills could create their own software solutions, perfectly adapted to their instant and unique requierements. These projects are currently subsidized by the EU and the Spanish Ministry of Industry, Tourism and Commerce. For the sake of clarity we have separated the authoring and runtime phases of the service front-end lifecycle (see Fig. 6).

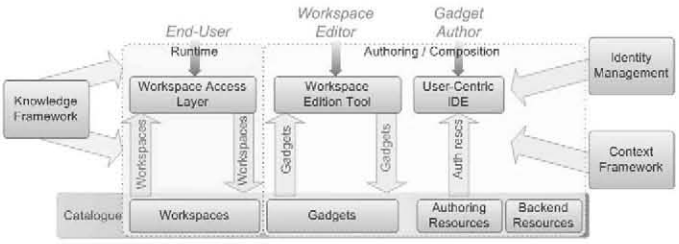

Figure 6. Proposed Architecture for Next-Generation Service Front-Ends (Overview) to Enable EUD

Gadgets will be the main building blocks of such an architecture. A gadget implements the user interface and application logic necessary to interact with one or more underlying services. Gadgets are self-contained front-end components focused on a single goal and, consequently, are of limited complexity. Gadgets can be grouped into workspaces.

Our proposed architecture for the authoring phase includes two main components:

- A "Gadget Authoring Tool", which it is a user-centric IDE dedicated to gadget design and creation. This is a visual tool that assists non-IT-aware users in creating their own service front-end resources [15]. Using this user-oriented IDE gadget, authors will be able to visually design, reuse and share gadget screens, flows and back-end resource compositions or connectors among others. Authors will easily compose a gadget from a series of building blocks (authoring resources) available in a palette. This palette is actually a specific view of the resource catalogue and can contain UI artefacts (screens), operators, screenflows, ready-to-use back-end resources and compositions, etc.
- A "Workspace Editing Tool" intended to design custom user workspaces, as a mashup editor. This tool will permit the visual design, reuse and sharing of user workspaces by selecting, connecting and composing the most suitable gadgets for dealing with a domain problem. The ultimate aim is to create new, modular and anticipated service front-ends (instant applications) by combining smaller pieces (gadgets). Each user can have and share any number of workspaces with other members of the community.

These two tools will be supported, at least, by the following formalisms:

- A Declarative Authoring Language for describing device and modality-independent user interfaces. Traditional user interface development approaches are insufficient for supporting the new-generation service front-ends. Taking one step further, traditional UI platforms

and toolkits lack the formalisms necessary to deal with context-aware service front-ends. For example, there are no declarative mechanisms to specify how an interface should adapt according to different delivery contexts. Instead the developer needs to do the adaptation manually, using an ad-hoc and costly approach that does not promote reuse or standardization. We propose a layered approach to the development of UI for the services front-end: abstract UI (device independent), concrete UI (device dependent) and rendering for specific devices. All the layers can be represented in XML and embody a model of behaviour at a gradually finer level of detail.

- A standard format and infoset for the description of gadget metadata (Gadget Template). This template is a machine-readable gadget description that should at least contain information about author names and affiliations, date, a human-readable description, pointers to the gadget source code and, most importantly, publish-subscribe metadata depicting what data items the gadget publishes (including their type, name, semantics, etc.) and what data items are consumed by the gadget (including their type, name, semantics, etc.). Finally, the template should contain context metadata about the gadget.

At runtime we propose a "Workspace Access Layer" that is responsible for giving end users access to one or more workspaces. The layer will be in charge of rendering each user's workspace (and the gadgets it contains) depending on the characteristics of the target context, adapting a workspace and the gadgets it contains to the restrictions dictated by the target context and also implementing all the artefacts needed to support the execution of gadgets at runtime, such as publish and subscribe communication mechanisms or persistence of gadget data and state, providing a runtime environment for gadget execution.

Additionally, there will be a set of horizontal modules dedicated to different aspects that are common to the runtime and authoring phases and that were explained in Section III-E:

- A Resource Catalogue containing all the metadata about the different building blocks of the architecture (gadgets, screens, flows, workspaces, content delivery resources, application data resources, resource compositions, etc.).
- A Context Framework implementing all the concepts, formalisms and artefacts described previously in this paper.
- Identity and Session Management for dealing with user session and identity among others.
- A Knowledge Framework following the approach described in Section III-E of this paper.

Figure 7. EzWeb platform

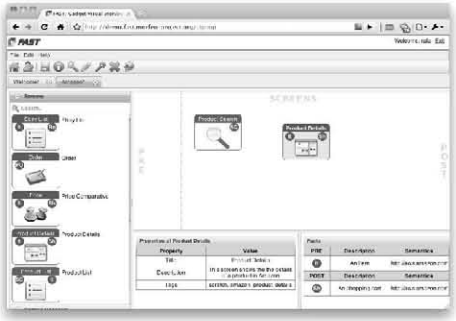

Figure 8. FAST IDE

V. Implementation and validation of the proposed architecture

The proposed architecture was implemented on the EzWeb/FAST framework. This framework is composed of two tools derived from the EzWeb[2] (Figure 7) and FAST projects[3] (Figure 8). Both tools are publicly available at http://demo.ezweb.morfeo-project.org and http://demo.fast.morfeo-project.org, respectively.

EzWeb is a mashup platform where gadgets could be interconnected and arranged in several workspaces to satisfy instant requirements. Multi-purpose gadgets are published in a collaborative catalogue. However, if end users are unable to find what they need, they can easily create new gadgets using the FAST development environment, designed to enable non-technical users to create gadgets from more specific components called resources, available in public catalogues and around Internet. A short video, available at http://www.youtube.com/watch?v=qFt2LBlxkwU (part 1) and http://www.youtube.com/watch?v=dpoRhnF8_1A (part 2), demonstrates EzWeb/FAST features and teaches end users how to compose their own applications. Figure 9 illustrates the EzWeb/FAST composition model that strictly implements the reference architecture proposed in Section IV.

[2]Morfeo EzWeb, http://ezweb.morfeo-project.org
[3]Morfeo Fast, http://fast.morfeo-project.eu

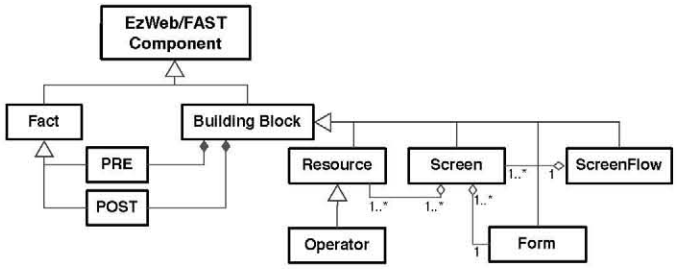

Figure 9. EzWeb/FAST composition model

Now that we have presented the EzWeb/FAST composition model, research focuses on evaluating its use and proving that our premise of enabling end users with no programming skills to build their own composite applications is feasible and true. For the evaluation we used the existing EzWeb/FAST framework that strictly implements the proposed architecture. If this framework is validated, then we will also have validated the underlying architecture model. EzWeb/FAST evaluation aims to test whether the developed user-centred composition system satisfies its usability, functionality and performance requirements. We present some of the findings of the research we conducted within the 2009/2010 FAST project reporting period [16].

In the above report we took a holistic approach to EzWeb/FAST system analysis from the information systems research viewpoint [17]. This perspective focuses on solving practical problems in the interaction between the organisation, people and information technology. Consequently, we conducted the research from the organisation, user, and information technology angles, aiming at gathering and structuring feedback about both the EzWeb/FAST system and the underlying architecture as a basis for improvement.

We experimented with three different approaches for running the holistic study, thus covering all the evaluation perspectives:

- An **expert evaluation** from a business consultant perspective. The expert was familiar with traditional business process modelling approaches, and represented the EzWeb/FAST system target user group.
- A **case study** as a strategy of empirical enquiry representing a specific real-life situation. It addresses the organisation and IT perspectives.
- A **laboratory experiment** under controlled conditions, covering the user perspective.

Based on the expert evaluation approach, we identified three key findings regarding the improvement of the EzWeb/FAST service composition system usability:

- **Design and runtime convergence**. This involves the composition of real-time data by non-experts. Composite applications are built from real data sources, consequently traditional test systems will have to evolve in new ways to support this convergence.
- **The organisation is similar to communities**. User-centred composition environments are structured similarly to a community. Building block sharing within the community is a key issue.
- **The need for user guidance**. Our composition system aims at supporting consultants in their daily work. Consultants usually have limited programming experience. This point uncovers the fact that users need guidance to create composite applications and extensive sample libraries in order to reuse existing solutions.

From an organizational perspective, the case study showed that the use of user-centred service composition systems has a number of interesting business benefits. The most important findings were that the use of EzWeb/FAST leads to higher employee productivity, as they can use the time they save to complete other tasks. It also improves user flexibility, involvement and satisfaction.

Last but not least, we conducted a laboratory experiment which accounted for the user perspective. This was the core and most important part of the evaluation, because it is the users that determine the success of the community- and user-oriented service composition paradigm

No.	Question
	Usability
Q1	FAST was easy to use first time round
Q2	I would imagine that most people would learn to use FAST very quickly
Q3	I felt confident using FAST
Q4	I didn't need to learn a lot about FAST before I could use it effectively
	Functionality
Q6	Screens were easy to find
Q7	The screenflow of a composite application was easy to model
Q8	I had no problem defining inputs and outputs
Q9	The designed composite applications were easy to publish
	Performance
Q14	The FAST system responded too slowly to inputs
Q15	The system ran stably
	General
Q16	I was able to set up screens for the B2B scenario
Q17	The evaluation task was too difficult

Table I
MAIN QUESTIONNAIRE QUESTIONS

One of the tasks set for the experiment participants was to create parts of a complex and real-world B2B scenario. To do this, they used the EzWeb/FAST components (i.e. *screenflows*, *screens*, *gadgets*, and so on.). This way, we were able to evaluate design and runtime convergence. They also had to fill in a *questionnaire* stating their individual opinion and impression of the EzWeb/FAST system. In a semi-structured *focus group*, they were asked about their attitude to EzWeb/FAST. And, apart from this, they were observed doing the main evaluation task. In sum, we applied three data collecting instruments (both quantitative and qualitative) to ensure that the quality of the results of

the evaluation was beyond all question and to assure that the evaluation had a solid basis. The results of this user evaluation are summarized below.

We received feedback from a total of 41 participants, where 21 had little or no programming experience —business users—, whereas the other 20 were expert programmers —technical users. The results reveal that users have a positive impression of the EzWeb/FAST system. Figure 10 represents the statistical feedback based on the questionnaire data. The questions are listed in Table I. The EzWeb/FAST usability was rated as neutral with a positive tendency. We observed that participants found the components that they needed and did not have to do a lot of learning to use EzWeb/FAST. From a functionality perspective, we found that participants had some difficulties in finding the right screen to use. Additionally, input and output definitions were not self-explanatory. On the other hand, users found the procedure for publishing designed composite applications on a target platform easy to follow. Regarding the performance, EzWeb/FAST was stable and there were no critical incidents throughout the whole evaluation time frame.

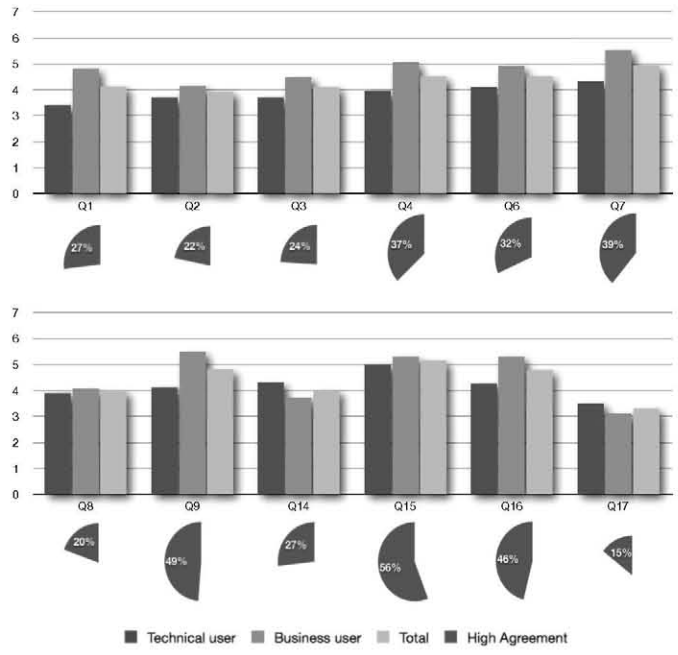

Figure 10. Questionnaire about EzWeb/FAST

Human Factors	Value	Adjusted Value
Perceived Usefulness	2	2,24
Computer Enjoyment	2	1,86
Subjective Norm	2	1,46
Perceived Ease of Use	2	1,3
Network Externality	2	0,96
Task-Technology Fit	2	0,72
Computer Self-efficacy	1	0,35
Internal Computing Support and Training	1	0,16
Total		***9,46***
HCI Factors	**Value**	
Abstraction Gradient	2	
Closeness of Mapping	2	
Consistency	2	
Diffuseness	2	
Error-proneness	1	
Hard Mental Operations	2	
Hidden Dependencies	1	
Premature Commitment	2	
Progressive Evaluation	2	
Role-Expressiveness	2	
Secondary Notation	1	
Viscosity	2	
Visibility	2	
Total	***8,84***	
Specialization/Functionality Factor	**Value**	
Total	***High***	

Figure 11. EzWeb/FAST EUD success evaluation based on validation study

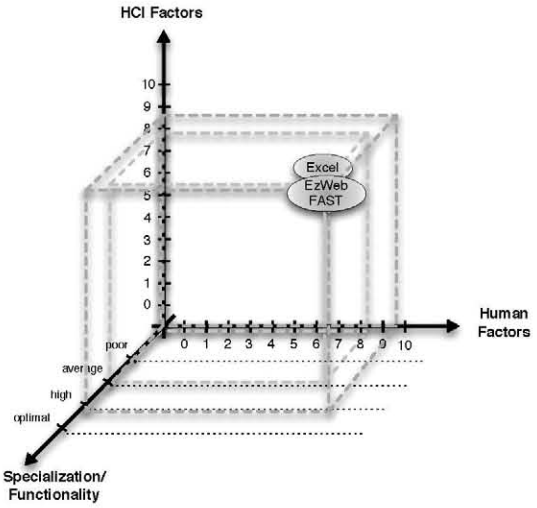

Figure 12. EzWeb/FAST EUD success represented in the proposed framework (and compared to Excel)

Looking at the overall impression of EzWeb/FAST, it is noteworthy that more than 70% of the participants rated EzWeb/FAST as good or excellent, recognizing its EUD potential. A comparison of the impressions of technical and business people revealed an interesting point: business people are more enthusiastic about EzWeb/FAST than technical users. A possible explanation for this is user empowerment. These results indicate that users with no programming skills are able to create composite applications on their own, and, consequently, demonstrate that our composition model is valid too.

Finally, EzWeb/FAST was evaluated following the EUD success framework presented in the Section III-D. End users were asked about human, HCI and specialization/functionality issues, and Figure 11 presents the results of the questionnaires.

Analysing these data, EzWeb/FAST can be compared with other successful EUD solutions, like Excel (Figure 4), where it has proved to be even more suited for end-user requirements. Therefore, it should be successful for EUD (Figure 12).

VI. Conclusion

The first effect of the advent of user-centric approaches for constructing next-generation service front-ends such as the one proposed in this paper will be to unleash unprecedented potential with respect to the consumption of electronic services by different stakeholders. As a result, large enterprises will be able to capitalize on faster application development (thereby reducing application development backlogs in IT departments), a more agile system landscape, and the empowerment of their employees to contribute to the design of the applications that they are supposed to use. Small and medium-sized enterprises will be enabled to select and compose resources hosted by a wealth of third parties rather than paying for pre-determined, inflexible and potentially heavyweight solutions. Finally, private individuals will benefit from intuitive, unsophisticated ways to discover, remix and use the Web-based services that they consider interesting and useful to build a EUD environment based on a user-centred ecosystem of services.

Besides the discussed benefits for different user groups, the novel user-centric approach will also abet the large-scale proliferation of what is often referred to as the Internet of Things (IoT). Not until information gathered from the multitude of dispersed sensors is made accessible and usable through an agile service front-end architecture will the envisioned Internet of Things become reality.

Acknowledgment

This work was supported in part by the European Commission under the first call of its Seventh Framework Program (FAST STREP Project, grant INFSO-ICT-216048), the European Social Fund and the UPM under its researcher training programme.

References

[1] G. Alonso, F. Casati, H. Cuno, and V. Machiraju, *Web Services Concepts, Architectures and Applications*, ser. Data-Centric Systems and Applications. Germany: Springer, 2004.

[2] C. Schroth and O. Christ, "Brave new web: Emerging design principles and technologies as enablers of a global soa," in *Proceedings of the IEEE International Conference on Services Computing, 2007. SCC 2007.* Los Alamitos, CA, USA: IEEE Computer Society Press, 2007, pp. 597–604.

[3] D. Lizcano, J. Soriano, M. Reyes, and J. J. Hierro, "Ezweb/fast: Reporting on a successful mashup-based solution for developing and deploying composite applications in the upcoming "ubiquitous soa"," *Mobile Ubiquitous Computing, Systems, Services and Technologies, International Conference on*, vol. 0, pp. 488–495, 2008.

[4] A. P. McAfee, "Enterprise 2.0: The dawn of emergent collaboration," *MIT Sloan Management Review*, vol. 47, no. 3, pp. 21–28, 2006.

[5] ——, "Will web services really transform collaboration," *MIT Sloan Management Review*, vol. 46, no. 2, pp. 78–84, 2005.

[6] R. M. Balzer, "Imprecise program specification," *Report ISI/RR-75-36, Information Sciences Institute*, December 1975.

[7] M. D. McIlroy, "Mass produced software components," in *Software Engineering, Report on a conference sponsored by the NATO Science Committee, Garmisch, Germany*, October 1968, pp. 138–155.

[8] J.-H. Wu, Y.-C. Chen, and L.-M. Lin, "Empirical evaluation of the revised end user computing acceptance model," *Computers in Human Behavior*, vol. 23, no. 1, pp. 162 – 174, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/B6VDC-4CF5BTH-2/2/1c5438462f07b8c0745efb54bee4fed1

[9] S. P. Jones, A. Blackwell, and M. Burnett, "A user-centred approach to functions in excel," in *In ICFP 03: Proceedings of the eighth ACM SIGPLAN international conference on Functional programming*. Sweden, EU: ACM Press, 2003, pp. 165–176.

[10] H. Lieberman, F. Paternò, M. Klann, and V. Wulf, *End-User Development*, ser. Human-Computer Interaction Series. Germany: Springer, Nov. 2006, vol. 9, ch. End-User Development: An Emerging Paradigm, pp. 1–8.

[11] F. D. Davis, R. P. Bagozzi, and P. R. Warshaw, "Extrinsic and intrinsic motivation to use computers in the workplace¡sup¿1¡/sup¿," *Journal of Applied Social Psychology*, vol. 22, no. 14, pp. 1111–1132, 1992. [Online]. Available: http://dx.doi.org/10.1111/j.1559-1816.1992.tb00945.x

[12] T. Green and M. Petre, "Usability analysis of visual programming environments: A cognitive dimensions framework," *Journal of Visual Languages and Computing*, vol. 7, no. 2, pp. 131–174, 1996.

[13] R. Smith, "Enterprise mashups: an industry case study," in *Keynote at New York PHP Conference and Expo*. NY, USA: IBM Software Group Press, Jun. 2006.

[14] C. Anderson, *The Long Tail: Why the Future of Business Is Selling Less of More*. NY, USA: Hyperion, July 2006.

[15] R. T. Fielding, "Architectural styles and the design of network-based software architectures," Ph.D. dissertation, University of California, Irvine, 2000.

[16] V. Hoyer, A. Fuchsloch, S. Kramer, K. Moller, and J. López, "Evaluation of the implementation," FAST Consortium, Tech. Rep. D6.4.1, February 2010.

[17] A. S. Lee, "Remarks from mis quarterly editor - inaugural editor's comments," *MIS Quarterly*, vol. 23, no. 1, 1999.